com\BankersAlgo.java

```java
import java.util.Scanner;

public class BankersAlgo {
    public static void main(String[] args) {
        Scanner cin = new Scanner(System.in);
        System.out.println("Enter number of processes : ");
        int n = cin.nextInt(), count = 0;
        boolean[] finish = new boolean[n];
        System.out.println("Enter number of resources : ");
        int m = cin.nextInt();
        int[][] allocation = new int[n][m];
        int[][] max = new int[n][m];
        int[][] need = new int[n][m];
        int[] avail = new int[m];
        int[] sq = new int[n];
        System.out.println("Enter number of Instances or resources avaiable for resourcej");
        for (int i = 0; i < m; i++) {
            System.out.print("Enter available of resources" + (i + 1) + " : ");
            avail[i] = cin.nextInt();
        }
        System.out.println("Enter resources already allocated for each process ");
        for (int i = 0; i < n; i++) {
            finish[i] = false;
            for (int j = 0; j < m; j++) {
                System.out
                        .print("Enter resources" + (j + 1) + " already allocated for process " + (i + 1) + " : ");
                allocation[i][j] = cin.nextInt();
            }
        }
        System.out.println("Enter Maximum instances of each resouces required for each process");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                System.out
                        .print("Enter Max resources" + (j + 1) + " required for process " + (i + 1) + " : ");
                max[i][j] = cin.nextInt();
                need[i][j] = max[i][j] - allocation[i][j];
            }
        }
        while (count != n) {
            for (int i = 0; i < n; i++) {
                boolean y = false;
                if (finish[i] == false) {
                    for (int j = 0; j < m; j++) {
                        if (need[i][j] <= avail[j])
                            continue;
                        else {
                            y = true;
                            break;
                        }
                    }
                    if (y == false) {
                        for (int j = 0; j < m; j++)
                            avail[j] += need[i][j];
                        finish[i] = true;
                        sq[count] = i;
                        count++;
                    }
                }
            }
        }
        System.out.print("Safe sequece : ");
        for (int i = 0; i < n; i++)
            System.out.print("p" + sq[i] + " ");
        cin.close();
    }

}

```